

# For a few dollars less

**FPGA VENDORS ARE LINING UP TO OFFER DENSE LOGIC PARTS AT PRICE POINTS OF JUST A FEW EUROS, AND CAPABLE DESIGN SOFTWARE SUITES THAT ARE EITHER FREE OR NEARLY FREE—TIME TO TAKE A SECOND LOOK AT WHAT YOU MIGHT PACK INTO A PROGRAMMABLE PART?**

If the opening paragraph of this article had set out its scope and subject material as “System-on-Chip”, there is a good chance that a proportion of the readers it is aimed at would already have dismissed it and moved on to some of the other content of this issue of EDN Europe. The terms System-on-Silicon or System-on-SHIP (SoC) have acquired a certain amount of baggage: vendors of products such as large-scale ASICs have used them so freely that “SoC” has become associated with mega-gate designs. Products such as cellular-handset baseband chips, or mixed-technology designs such as Bluetooth transceivers that combine logic and RF on a single piece of silicon, fit this profile. Along with the fact that they embody millions of logic gates, their designers must target manufacture on leading-edge semiconductor processes, with

all that that implies: extremely complex design processes, very high up-front design costs and huge production volumes to achieve economic viability. If your normal operating domain is modestly-sized micro-processor or microcontroller-based designs that you will build in modest—or even very small—volumes, you might very well dismiss the topic as being of no interest.

All those connotations that go along with SoC can obscure the fact that in today’s technologies, System-on-Chip can equally well mean small-system-on-chip, using parts from the newer families of FPGAs. As FPGA vendors introduce each new range of parts, those that draw most attention are the top-of-the-line, most complex ones. The same technology that enables mega-gate programmable parts at the top of the line also delivers extremely capable but less complex parts, at prices of just a few euros, in the same product family.

## FPGAs GO DEEP SUB-MICRON

Several years ago, when we in the media reported on progress in programmable chips, we tended to focus on the architecture of the programmable logic itself. To make a working programmable logic chip takes a lot of extra circuitry over and above the logic itself. To all the pass transistors that select the desired logic configuration, you must add more of the same that connect the logic to wiring segments, and even more to connect together the logic and in turn connect it to the I/O pads and the outside world. You have to supply sufficient wiring—“routing resource”—to not only

cater for all of the logic functions you wish to implement, but also to interconnect all of the configuration circuitry. In short, there’s a lot of “stuff” on an FPGA die in addition to the logic that builds the functions you want to end up with. In older technologies—0.25-micron and larger (earlier) silicon geometries—packing in sufficient amounts of logic was a challenge for the chip designer, and so was conceiving a basic architecture that could handle all of the likely demands on the chip without running out of routing resources.

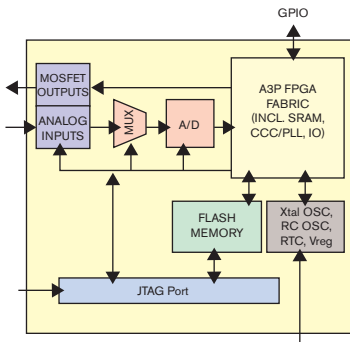
When foundries began to offer high-volume silicon processes at 0.13 micron and, now, 90 nm, the FPGA landscape changed. Designers could, in the architecture of their choice, make logic available with system-level gate counts, without taking up too much silicon area—even including the allowance for the ancillary “stuff”. As ever, silicon area equals cost. And, those processes come with lots of metal layers (the equivalent of PCB layers at a board-design level). Today’s FPGA designers employ six, seven or more such layers and are largely free of earlier restraints on routing. As well as interconnecting logic, they can add features such as logic-analyser-style access to internal device nodes. Vendors designed earlier devices prioritising the amount of logic each one incorporated. Then, they added I/O to suit. In today’s product ranges, they are likely to define the smaller devices by the package and the amount of I/O on offer. Virtually all of today’s FPGA offerings provide I/Os that are configurable to a wide range of signal

levels and standards—but those I/O pads don’t scale down in proportion with the logic itself as process geometries shrink. Having defined the device by an I/O count that matches one or the other market, the vendor then fills the chip with as much logic as space allows. In the latest device technologies, that can be a lot.

If you have a design that involves a microprocessor or microcontroller, memory, and the usual peripherals—one that is

### AT A GLANCE

- \* Sufficient programmable logic resources to absorb a typical small/medium  $\mu$ C-based system now come with a price tag of just a few euros.
- \* Development systems begin with free versions: expect to pay a few hundred euros for a more capable version.
- \* All silicon offerings come with a mix of free (bundled) IP and chargeable, third-party-sourced IP.
- \* Design styles vary: HDL knowledge will help, but you can remain in a prototype-orientated mode of operation for much of the design process.
- \* Familiar processor cores are available: maximum flexibility for least money comes with conversion to the FPGA vendors’ in-house cores.
- \* Benefits include system upgradeability, protection from component obsolescence and simplified PCB and hardware design.
- \* In this article there is no discussion of performance; 90-nm-technology chips will all run at hundreds of MHz, and clock speed is unlikely to be a limitation in the context of integrating  $\mu$ C-based designs.



**Figure 1** With analogue inputs, flash memory and MOSFET driver outputs, Actel's Fusion PSC can extend the single-chip concept to cover even more of a control system.

a complete system in its own right, or that is itself a sub-system within a larger design—it is both feasible and economic to aim for a single-chip programmable solution. “Single-chip” requires qualification in that any analogue functions in the design will be external to the FPGA, although with a device that Actel recently introduced, even that is not necessarily true. A reduction in printed-circuit-board area and complexity will aid the economics of a single-chip solution.

Market leaders Altera and Xilinx both have a twin-track approach to their leading-edge product ranges. Both produce a full-featured, latest-technology series of parts, namely Virtex (Xilinx) and Stratix (Altera). For designers who do not require all of the most advanced features, both companies assume that a range of parts that omits certain attributes will satisfy a large segment of market demand while being much less expensive to produce and sell. Xilinx has Spartan, and Altera its Cyclone ranges. It is the smaller parts in these series that potentially provide the designer with “most bang for the least bucks”.

At Altera, European marketing manager Pat Mead makes the point that there is an effective “floor” to the pricing that you can expect, that the packaging sets. If you are looking to replace a microcontroller-based circuit with a single chip, it's likely you will need a reasonable number of I/O lines plus, perhaps, external bus connections. The package will absorb a significant proportion of the pricing of the smallest devices in all of the families mentioned below—so you need to make a careful assessment of your real I/O needs.

Cyclone II is a cost-optimised family of SRAM-based FPGAs from which, Mead says, you might consider the first two parts—the EP2C5 and 2C8—in the role of replacing a small microcontroller-based

system with a single chip. Altera measures logic density by counting its logic elements; these two parts have 4608 and 8256 LEs, respectively, and about 120 or 165 kbits of RAM. There are also other hard-coded functions such as embedded multipliers to increase the area efficiency of implementing computational and DSP functions. As ever, there's an arbitrary factor to take into account when trying to equate those figures with logic gates in the ASIC domain; more useful, Mead suggests, is to consider what you can get into one package if you set about the single-chip concept. At this point, you need to consider that if you are replacing a microcontroller or micro-processor, you will need just such a core. You can acquire the source code (HDL) for a standard microcontroller core, such as the 8051 in one of its many variants, and compile that into the FPGA; or you can use Altera's own processor, the Nios II. Altera designed the Nios architecture with programmable-logic implementation in mind, making it more efficient than other IP-sourced processors in terms of FPGA resource usage—this, Mead says, can make a significant difference in overall cost-effectiveness when using a small device. Nios is a configurable and extendable design that will provide up to 100 DMIPs performance in Cyclone II. A minimum configuration of the part would run at about 15 DMIPs, and Altera says that a 2C5 will take one such core plus an Ethernet MAC, HDLC controller, UART, SPI interface, PIO and timers. A 2C5 will be priced around \$3, in 100k quantities, in 2006. (If your needs are more modest, scale that estimate upwards accordingly.) You might also consider devices from the original Cyclone family—the 1C3, 1C4 and 1C6—for a small system, if cost is critical. But Mead again cautions that packaging costs dominate at these low price points, and you need to be careful in balancing I/O needs and package options.

#### µP CORE ON- OR OFF-CHIP

There is a large variety of combinations that can be created to achieve a minimum parts-count system. You might retain a discrete microprocessor on the PCB, and pair it with an FPGA configured to provide all of the required peripherals; or, the same combination but including a “soft” processor core in the FPGA alongside the peripherals to handle specific functions; or the true single-chip approach, with a soft core and all peripherals on the FPGA. Altera offers a software product called SoPC (System-on-Programmable-Chip) Builder

that configures the peripherals in a largely automatic fashion. With a core—such as the Nios—taking up only a fraction of the logic in the smallest device in the family, you can also contemplate a multi-processor solution, with different parts of the applications partitioned among multiple instances of a processor core. The 2C5 will house five minimum-configuration Nios IIs, Mead says. The motivation for such a design might be to achieve higher performance—or it could be to reduce power, using multiple processors each running at a much reduced clock rate.

Altera's software package, Quartus, comes in a number of bundles beginning with a free downloadable version. A single payment of \$995 buys a royalty-free licence for Nios and the tool set allowing designers to develop software.

Xilinx's low-cost Spartan series of parts also employs 90-nm silicon technology for the highest logic density. Its designers have addressed the issue of balancing I/O with logic density by offering alternative ranges—the base Spartan 3 devices are optimised for I/O count, while the 3E series provides the maximum amount of logic for a given silicon area. The two smallest devices in the 3E family—the 3S100E and 3S250E—have 2,160 and 5,508 logic cells respectively; Xilinx equates this to capacities of 100k and 250k “system gates”. In the Spartan architecture there are blocks of RAM as well as distributed RAM bits (logic resources that can be allocated as RAM): the 250E can have up to about 250 kbits of RAM. Again, a selection of packages allows the user to optimise I/O count and package costs at each device level. Once again, pricing for the families' smallest devices is in the range of “a few” dollars or euros. Xilinx quotes \$2 for the 100E, for 500k quantities, in the second half of 2006—again, you will need to reckon on the usual factors to reach small-volume pricing. Even if you have to scale the price up to distribution-level, small-quantity pricing, a few euros still buy a device that will contain a significant system: some of the embedded features include hardware for DSP support and as is virtually standard, configurable I/O that directly interfaces to most common bus and signaling conventions. Flexible I/O in itself greatly simplifies the task of implementing a single-chip system—when you can directly connect the chip to external buses, the need to design-in the physical layer for external connections largely disappears. Xilinx's Dublin-based marketing manager

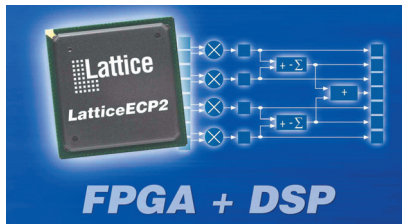
John Dillon notes that when using embedded soft processor cores, the opportunity for hardware/software co-design increases: you can add hardware features to reduce the load on an embedded processor core. He points to features such as Xilinx's FSL (Fast Simplex Link), that provides high-speed dedicated data transfer links, speeding up throughput on critical nodes of a design. Xilinx's "own" embedded core is the MicroBlaze, although you can of course acquire and embed the IP for any other core of your choice. (There is also a very small 8-bit core in the same family, the PicoBlaze.) Dillon notes that by adopting and becoming familiar with MicroBlaze, users multiply opportunities for hardware/software re-use and also gain some protection against obsolescence, as Xilinx is committed to continuing and compatible support of the architecture on future hardware.

As with other vendor's ranges, Xilinx is aiming to reduce the up-front costs of starting work with its low-cost architecture to nominal sums. There is a free, downloadable development environment (ISE), and a basic "hard" kit costs \$100. Getting into MicroBlaze starts with a \$495 price-point version of the EDK kit. As with the pricing, so with the design paradigm—a GUI/Wizard-style interface guides you through configuring the core with no HDL programming required. A relatively standard MicroBlaze build will require around 1000 logic cells, Dillon says, so with even the smallest chip in the 3E series, "you have half the device to do other things".

With SRAM-based FPGA architectures, although all of the application may be running within a single chip, there is still a need for at least one more device: the configuration memory. For low-cost systems, vendors have focused on providing the configuration memory and boot-up support at the lowest possible cost. Users can program the Spartan device via a serial SPI interface or a faster parallel path, using commodity parts to hold the configuration code for the lowest possible cost.

## FLASH FPGAs GAIN GROUND

An alternative approach is to employ a flash-based device with the configuration code held on the same die. The extra memory adds silicon area so for devices with the same nominal amounts of logic, expect to pay somewhat more—but you must factor in the elimination of the separate memory chip, and a further reduction in PCB complexity. There is also a security benefit, if that is important for your design.



**Figure 2** In its low-cost, 90-nm-technology ECP2 series, Lattice builds in hard-coded blocks with DSP multiply/accumulate and pipelining functions.

Actel has championed this approach: the smallest device in its ProASIC3 family is a 30k-gate part. The security aspect of being able to encrypt the configuration code has enabled Actel to offer ARM cores on a royalty-free licence basis, as reported in the May 2005 edition of EDN Europe (pg 10).

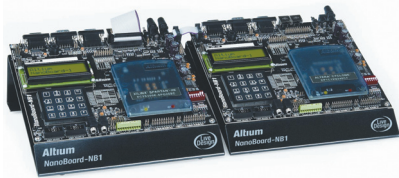
To this offering, Actel has now added a new variant: a configurable single-chip part that includes analogue signal handling (**Figure 1**). Based on a block of ProASIC3 FPGA logic, the chip adds an analogue-to-digital converter with multiplexer for signal acquisition and measurements, and MOSFET drivers for control outputs. Actel limits the analogue programmability to configuring and setting the parameters of these blocks as well as ancillary functions such as programmable-gain amplifiers—there is no programmable analogue in the sense of tile-arrays of basic analogue functions. Nevertheless, for control-loop-type applications, the "Fusion PSC" offers yet another option for a programmable-technology single-chip solution, at a "few-euros" price point. Once again, there is a low-cost (\$349) development starter kit to lower the barriers to beginning work with this part; and the chip itself, later this year, will sell for \$4.95 (250,000 unit volumes). See the January 2006 edition of EDN Europe (pg 20) for more details on the product.

At Lattice Semiconductor, European marketing consultant Bernie Perrin notes that integration of soft IP is a critical factor for success in the single-programmable-chip design space. The availability of macros for functions such as DDR memory interfaces allows the FPGA vendor to deliver blocks with tight timing requirements ready-to-use, leaving the designer free to integrate his unique "soft" content and potentially saving considerable design time. Lattice has parts in its XP, flash-based chip families that begin at around \$5, and in its low-cost EC families, down to \$2-3, Perrin states.

In the last few weeks, Lattice has introduced a new element to the System-on-Programmable-Chip sector in the form of its ECP2 family (**Figure 2**). Built on Fujitsu's 90-nm technology, this family spans logic array sizes from 6000 to 68,000 look-up tables (LUTs). Lattice quotes pricing levels as being "50 cents per 1000 LUTs", implying that the smallest device in the family will, once again, be in the \$3 region (here too, in high volumes.) The family builds in high-speed memory interfaces (DDR1 and 2) and, in keeping with Lattice's focus on communications functions, will interface directly to standards such as SPI4.2. A key reason for using a programmable system (and all vendors make the same point) is the option of extending the longevity of the design by permitting in-service software/firmware upgrades. The ECP family adds security for this process by including a dual-boot option. If a field upgrade should fail to complete properly, the chip will automatically revert to, and boot from, an alternate copy of the previous configuration, so that the system should never be left inoperable. In addition to 840-Mbit/sec I/O capability, the chips include hardware for DSP support with MAC/pipeline functions. In keeping with the market norms, there is a library of IP; and a low-cost route (including via download) into the ispLEVER tool suite that configures all of Lattice's FPGAs.

## BIND PCB AND FPGA LAYOUT

Once you have your single-chip design, you will need to place it on a board in the conventional manner. Alternatively, you could begin the complete design process on a software package that explicitly aims to ease the integration of FPGAs. This is the stated aim of Altium, which offers support in its PCB design suite that treats the FPGA as a component, and uses system-design techniques that are familiar from board-level design. The package allows a measure of device independence, treating the FPGA as a "black box" and mapping the design on to a specific device at a late stage. The Altium software comes with an IP library of pre-synthesised blocks such as processor cores and peripherals. The IP, which is royalty-free once the package is purchased, includes standard small processor architectures such as 8051, Z80 and PIC. There is also a 32-bit RISC "generic" core, the TSK3000. Designers can use the RISC core in its native form, or can employ a shell or wrapper as a means of providing hardware-level compatibility with other



**Figure 3** Linking two NanoBoard hardware prototyping “cartridges” mirrors a multi-core processing system mapped out in Altium’s design environment.

architectures such as Xilinx’s MicroBlaze and PowerPC, and ARM. Altium’s own compiler, Viper, provides C-level code compatibility between alternative processors. You can, the company says, use the TKS3000 for device-independent system development, then target the result to (for example) a Xilinx FPGA and the MicroBlaze core to yield a faster and more compact hardware solution.

The Altium design approach continues a strong parallel with a traditional hardware prototyping style using “Nanoboards”: these are FPGA daughter cards or cartridges for a development-board system, that allow users to load their software into FPGA hardware in a development environment, and to carry out conventional hardware-based debug. The approach supports multi-core processor development: multiple instances of a core within the software map to separate hardware (**Figure 3**) modules. Altium cites customer designs of up to seven linked cores (that example being for an unmanned-aircraft design).

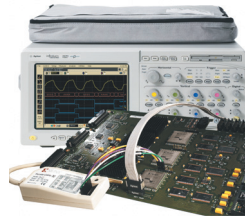
The same approach also supports a two-chip strategy, using a stand-alone microprocessor and an FPGA to implement all of the rest of the logic in a system. Using this model, you can work with the ARM architecture in the form of Sharp’s BlueStreak LH79520 chip and other devices in the ARM 7, 9 and 10 series. Altium will also look to support ARM within FPGA fabric in the form of the Actel devices. Altium’s European managing director Frank Hoschar notes that you can use a language-driven approach—the package embodies VHDL and Verilog engines—or you can use the pre-synthesised

IP that the package provides and build complex systems without much HDL knowledge. At the stage of mapping a design into a specific chip, you will need the FPGA vendor’s place-and-route software, but, as Hoschar points out, for smaller devices from most vendors’ ranges, the P&R software is either free or available at nominal cost.

Designs also benefit from a close integration of FPGA and PCB design processes, Altium says. Current-generation devices have great flexibility in their ability to internally map circuit nodes to I/O pins. The Altium software, in its version 6.0 form, has a “push-button” facility to couple that mapping to the PCB routing process—the software optimises the wiring rats-nest while looking back into the FPGA package. The package can produce greatly simplified pin-outs and PCB routes. With less need to “escape” signal paths out of complex BGA-device pin-outs, the software can complete the PCB design with fewer layers. Altium Designer 6.0 is available in three editions, ranging from a foundation edition that carries out all the front-end design processes—and that includes all the licensed IP—for just over € 4000. Adding full PCB implementation doubles that figure, and a comprehensive edition that includes software development tools is just under € 10,000.

### FPGA VIRTUAL PROBING

While working in the FPGA environment, you can also continue to develop and debug your circuit in a traditional “breadboard” style with product combinations such as Xilinx’s ChipScope Pro verification tools. At a device-configuration level, ChipScope software (“Core Inserter”) places small core modules into the FPGA that provide logic-analyser and bus-analyser access to any internal signals, using a system of virtual I/O that routes the captured data out through the chips’ JTAG programming port. ChipScope works in conjunction with oscilloscopes from Agilent’s Infiniium mixed-signal range, using an option that Agilent calls FPGA Dynamic Probing. Similar options are available for the Agilent 6000 series scopes, and for the company’s logic analysers (**Figure 4**). Infiniium MSOs (mixed-signal oscilloscopes) combine 16 logic-analyser-style channels with full-bandwidth ‘scope facilities. Using the option, each of



**Figure 4** Agilent’s FPGA probing option, together with Xilinx software, allows you to probe internal nodes of an FPGA as though it were a conventional logic PCB.

the 16 logic input channels can select from 64 possible internal nodes of the FPGA, including probing internal processor buses. Users can modify the parameters of the inserted cores in their designs, making changes such as switching from state to timing views, and modifying the accessed connections. Then, a re-build of the FPGA configuration sets the chip up for measurements at, the two companies assert, real-time or near-real-time speeds. Users can switch between pre-assigned connections within the FPGA without rebuilding the chip configuration, just as they would if they were probing a physical breadboard. From the Xilinx perspective, the package works with Virtex and Spartan parts, including the low-cost variants mentioned above. It requires Xilinx’s ISE design environment but again operates with any of its variants, including the evaluation version.

### THE ZERO-μP OPTION

In considering your single-chip solution, that bundles together the microprocessor and all its support logic, it may be worth asking whether you need the processor core at all. You used the processor in the first place because you needed to implement some complex logical behaviour, and writing code for a microprocessor was the simplest way of doing so. With the freedom to directly configure the logic in an FPGA, creating dedicated logic for just the functionality you require could be the smallest, cheapest or lowest-power route to a single-chip solution. Celoxica’s Design Kit DK offers one route to directly transforming algorithms in C into FPGA code: specifically, it supports the Xilinx and Altera families mentioned here. You can use it in a hardware/software co-design mode—or, you can generate logic alone and remove the software element from the single-programmable-chip equation altogether.**EDN**

### FOR MORE INFORMATION

**Actel**  
www.actel.com  
**Agilent**  
www.agilent.com  
**Altera**  
www.altera.com  
**Altium**  
www.altium.com

**Celoxica**  
www.celoxica.com  
**Lattice**  
www.latticesemi.com  
**Sharp**  
www.sharpsme.com  
**Xilinx**  
www.xilinx.com

You can reach  
Editor **Graham Prophet**  
at +44-118-935-1650,  
fax +44-118-935-1670,  
and gprophet@  
reedbusiness.com.

